# IBM WebSphere Portal Tools Update

- Where the newest tools fit
- Considerations for selecting tools

© 2013 IBM Corporation

In this deck we'll provide guidance on selecting from the available Portal tools:

- Script Portlet
- Digital Data Connector (DDC) and Social Rendering
- Web Experience Factory (WEF)
- Rational Application Developer (RAD)
- IBM Forms (Forms Experience Builder or FEB)

We'll start with a quick, prescriptive guide to choosing tools, and after that take you through a few slides on each tool and its features and benefits

© 2013 IBM Corporation

# Some useful criteria for comparing tools

Here are some useful criteria for evaluating and comparing tools:

1. **Skills**: What are the skills you have or want to grow in your team, and how do they match the primary skills used with the tool?
2. **Typical use cases**: What are your typical use case requirements, and how do they match the “sweet spot” of the tool?
3. **Deployment**: Would you prefer to deploy applications by putting JEE code on the server, or by updating a content repository or database?

© 2013 IBM Corporation

# Prescription for selecting a tool

- If you are building primarily data entry applications, FEB is a clear first choice
- If you are primarily listing, formatting, or displaying information, consider DDC, with its business user control and WCM-based rendering
- If you are building applications that are more general than those above, consider using Script Portlet, since it leverages widely available skills (HTML/JS/CSS) and it has all the benefits of WCM management and deployment (in context preview, projects, approval workflow, etc.)
- If you are most comfortable with JEE deployment or if you want to leverage additional features such as data access connectors, refer to the information on the benefits and features of WEF and RAD to find the best choice for your needs

© 2013 IBM Corporation

# Comparison of tools

| Tool | Primary skills involved | Typical use cases | Primary application code deployment |
|---|---|---|---|
| Script Portlet | JS/HTML/CSS<br><br>JS libraries and frameworks | Any type of application UI, using data from REST/JSON services<br><br>Client-side applications | WCM |
| DDC | WCM, HTML/CSS/JS | Lists and details views of external data – social, commerce, search, other data sources – with some update functions<br><br>Server-side rendering | WCM |
| WEF | WEF and Java, HTML/CSS/JS | Any type of application UI, with server-side data access support provided by WEF builders<br><br>Client-side or server-side applications or combination of both | Server WAR |
| RAD | Java and JEE, HTML/CSS/JS<br><br>Optionally, UI frameworks such as JSF and Spring | Any type of application UI, with server-side data access support<br><br>Client-side or server-side applications or combination of both | Server WAR |
| FEB | Some HTML/JS/CSS | Data entry forms | DB2 |

© 2013 IBM Corporation

# A short walk-through of the tools

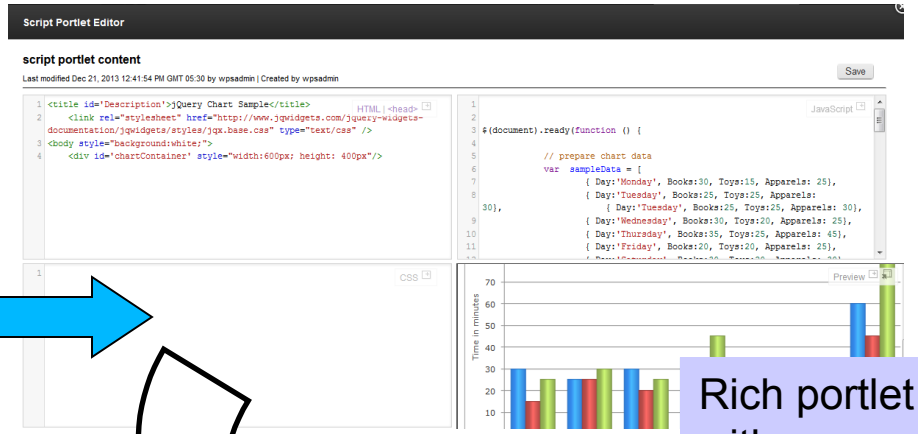
Script Portlet
Digital Data Connector (DDC) and Social Rendering
Web Experience Factory (WEF)
Rational Application Developer (RAD)
IBM Forms (Forms Experience Builder or FEB)

© 2013 IBM Corporation

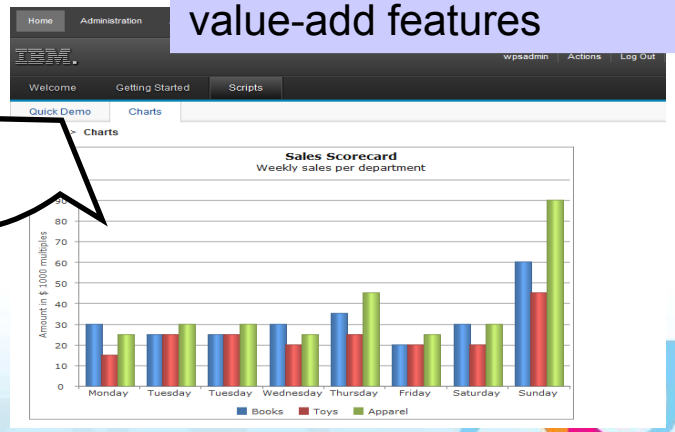
**Script Portlet** *Build JavaScript-based portlets using widely-available skills and libraries, in the cloud or on premises with no code deployment*

Application HTML/CSS/JS managed in WCM

JavaScript libraries and frameworks of your choice




Rich portlet applications with support for Portal value-add features




© 2013 IBM Corporation

# The Script Portlet approach

- All the application code – HTML, JS, CSS – is stored and managed in WCM
- From a browser you can work in the Script Portlet Editor, with syntax highlighting and auto-indent
- Applications can be initially developed using your tool of choice, then imported into Script Portlet
- Data access is done with Ajax/REST services using JSON data
  - These services can come from any provider, whether external via Portal's Ajax proxy or from WEF or other provider
- Portal's key value-add features are available
  - Inter-portlet communications, portlet preferences, responsive and adaptive design, Public Render Parameters, etc.

© 2013 IBM Corporation

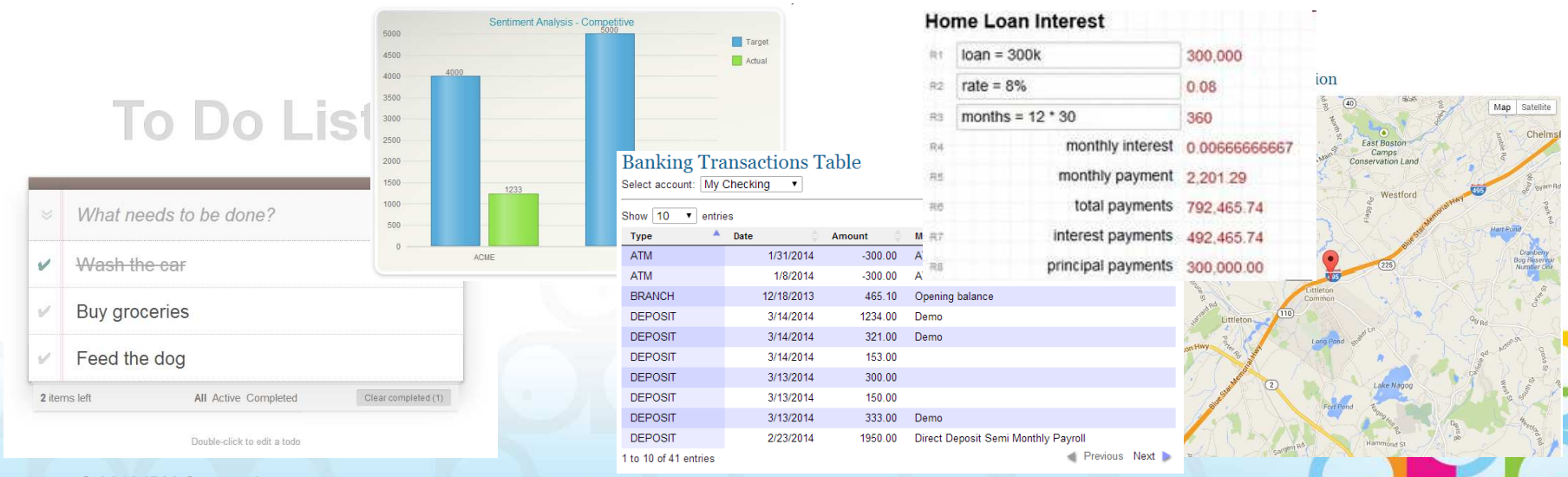
# Key benefits of the Script Portlet approach

- You can build portlets using skills you can more easily hire (JS/HTML/CSS, jQuery, etc.)
- Line of Business is less dependent on central IT - a small team of script programmers can make new portlets and customizations quickly and easily
- You can build portlets without deploying any new code on the server
  - Accessing a cloud server from a browser you can create a new portlet in minutes, without installing any tools, for example by copy/paste of JS and HTML snippets
- All the features of WCM can be used for managing applications
  - Projects and workflow can be used for approval by business and/or IT before applications go live
  - All coding changes can be scoped to the same project as other changes to web content or portal-managed content
  - You can create libraries of JavaScript, CSS, data, or markup, and link to them from within the WCM UI

© 2013 IBM Corporation

# Typical use cases for Script Portlet

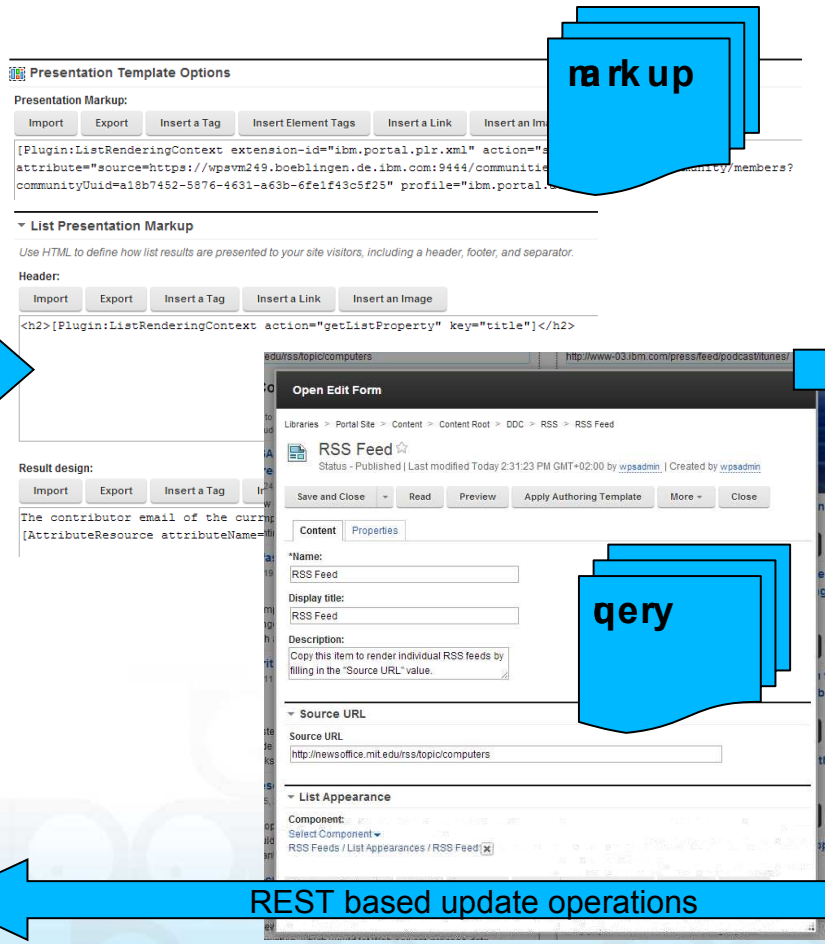
- Any type of application UI
- Forms, views, charts, etc., etc.
- Single-page applications
- Any JS libraries and frameworks can be used
- Data is accessed with REST/JSON services built with other tools or external
- Client-side applications




© 2013 IBM Corporation

# Digital Data Connector *Use DDC in WCM and WCM design components to render any external data, with little-to-no Java coding or deployment*

- IBM Connections
- WebSphere Commerce /Sterling Commerce
- Kenexa
- CMIS
- Omnifind/Search
- BPM (task list)
- Custom Feed/Anything



REST based update operations

# The DDC approach

- Use IBM Web Content Manager (WCM) design components to render data being served from external systems
- Provide a public plug-point to allow integration of all kinds of data
- Use WCM content items to describe what data to include and what design component to use to render the data
- Support dynamic context overwrites of source and filter attributes using public render parameters
- Make it easy to integrate remote XML data: No custom Java coding
  - Instead, use DDC list rendering profiles to transform external data into lists that can be rendered using WCM presentation templates
- Support update operations through WCM-generated user interfaces

© 2013 IBM Corporation

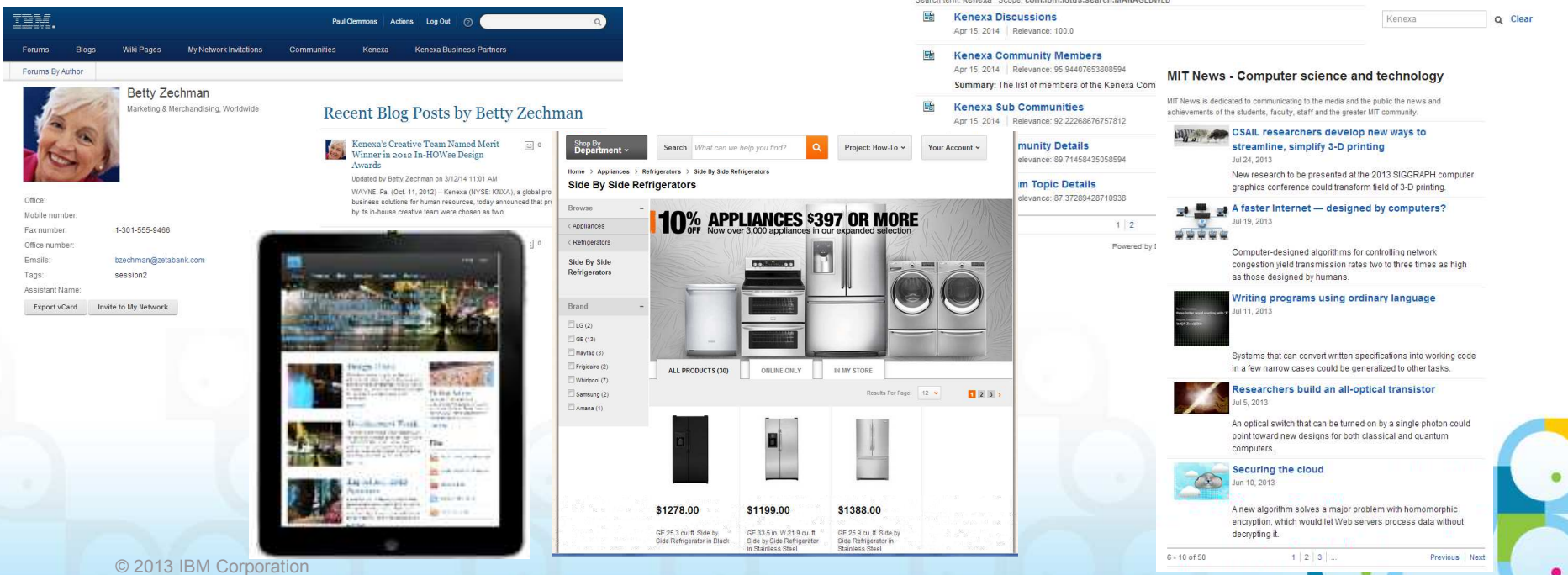
# Key benefits of the DDC approach

- WCM Designers have full control over list presentation
  → Resulting lists can fit nicely into the visual design of customer site
- No coding skills required for consuming & rendering external data
- Display any data on any device, with a consistent look managed using WCM
- Clean separation of concerns:
  - **Page Editors** use inline editing to: customize list definitions (i.e. what data to select, how to filter, how to sort) and to associate the appropriate styling by selecting it from a list of available list designs created by the Site Designers
  - **Site Designers** create the look and feel (markup and CSS styling) for the lists
- List presentations and designs support workflow, projects, preview, versioning, syndication
- Generic and shared caching and data retrieval infrastructure
- Can be further extended by developers
- No code deployment on the application server in many cases
  - DDC solutions often consist only of WCM artifacts (HTML/CSS/JS, WCM tags…)
  - Deployment can be done using WCM syndication

© 2013 IBM Corporation

# Typical use cases for DDC

- Lists and details display of external data using a WCM-driven look and feel
- Social, commerce, search, CMIS documents, feeds, or any other data sources
- Server-side rendering
- REST-based CRUD functions




© 2013 IBM Corporation

**Web Experience Factory (WEF)** *Use model-based code generation tools to quickly build portlets and generate multiple variations automatically*

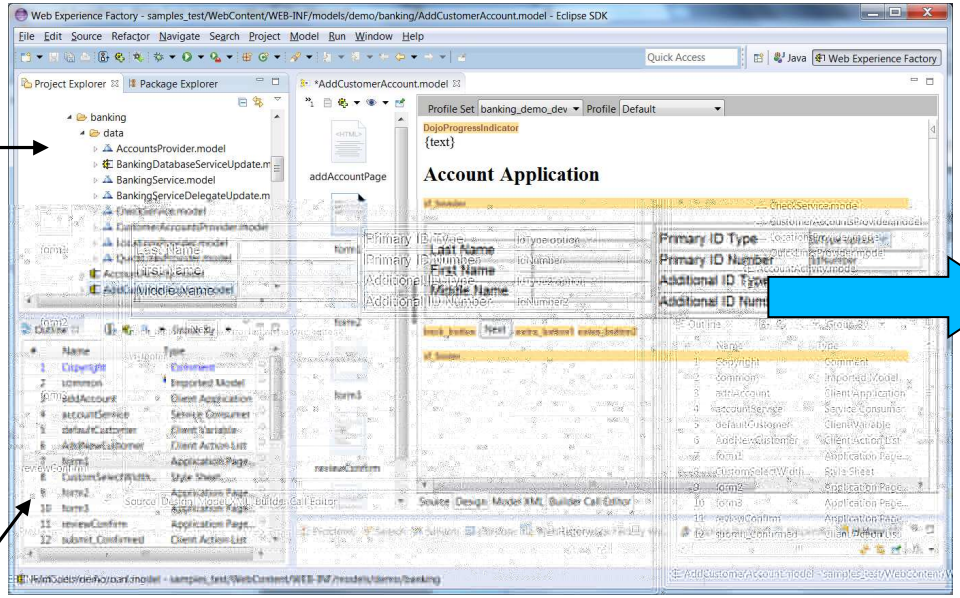
*Generated application code with multiple variations*

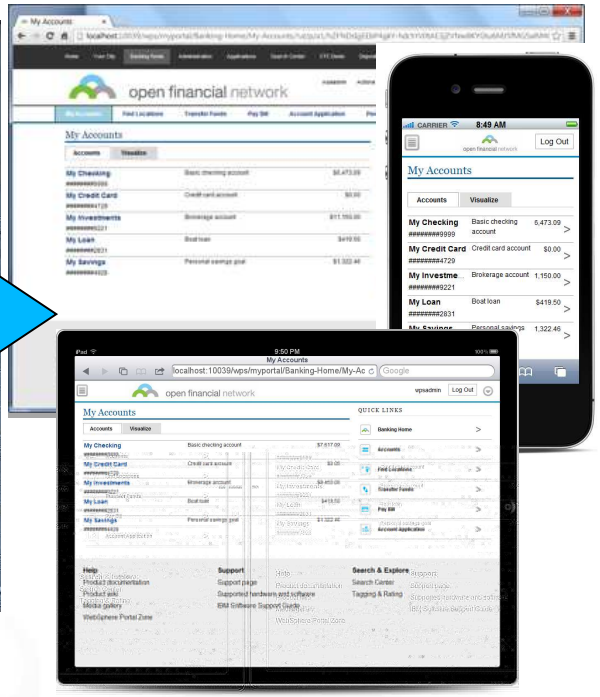
**Data Service Provider Model**

Data Access and Transformation Builders

**User Interface (Consumer) Model**

UI & mobile builders




© 2013 IBM Corporation

# The WEF approach

- Instead of hand-coding the application, developers configure builders in a model, and the builders generate the application code
- A large set of builders is available, for all sorts of tasks
  - High-level UI patterns, data access and transformation, mobile/responsive UI, rich data field behavior, Worklight support, etc.
- The Eclipse-based IDE includes a WYSIWYG view of the generated application
- Dynamic Profiling can be used to generate multiple variations from a single model
  - For different roles, device classes, geographies, runtime customization, and more
- Developers can publish directly to local or remote server

© 2013 IBM Corporation

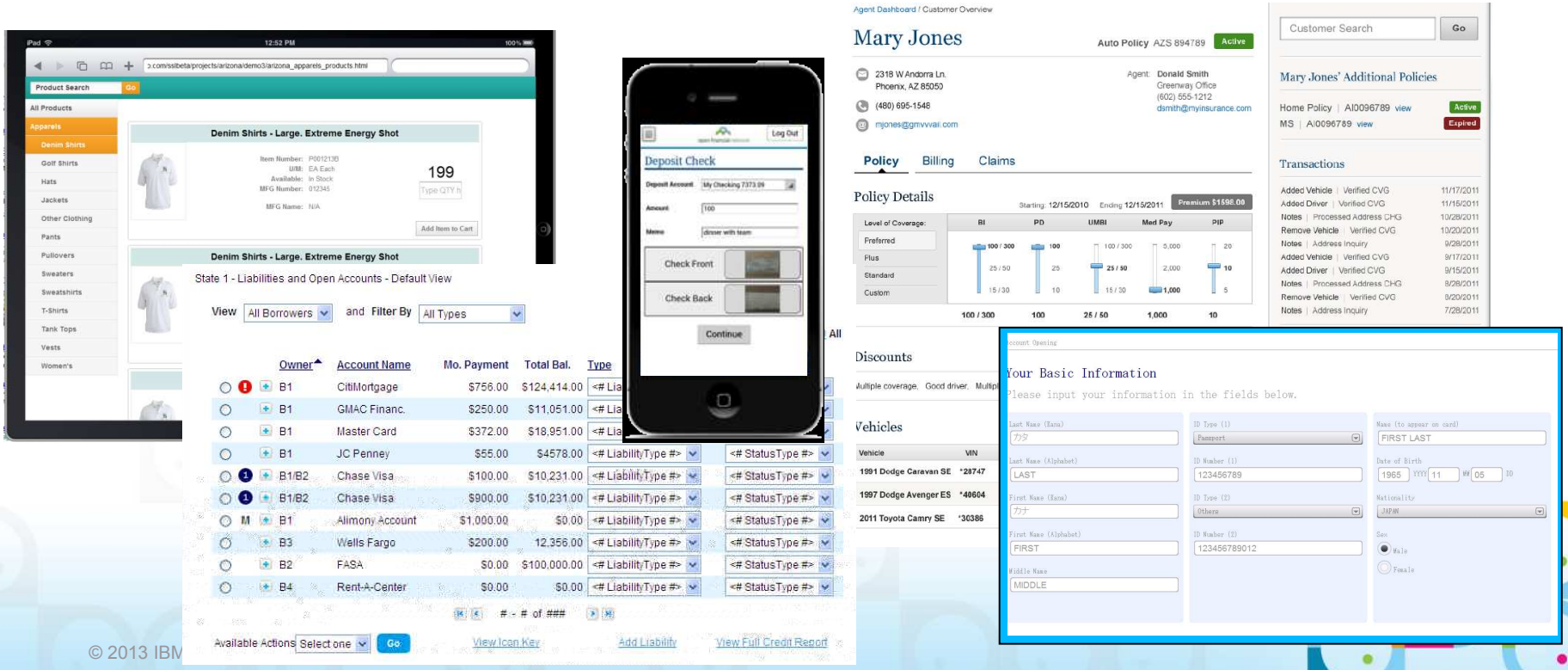
# Key benefits of the WEF approach

- Very rapid development of common patterns
  - Quickly generate fully-functional applications based on a service or back end data source
- No coding or learning curve for using complex technologies
- Easy integration with back end data and services using builders
- Full customization using builders and by integrating custom code and UI designs
- Flexibility and runtime variability using Dynamic Profiling
- Automatic enforcement of team development standards and practices
- Easily use a combination of server-side and client-side code

© 2013 IBM Corporation

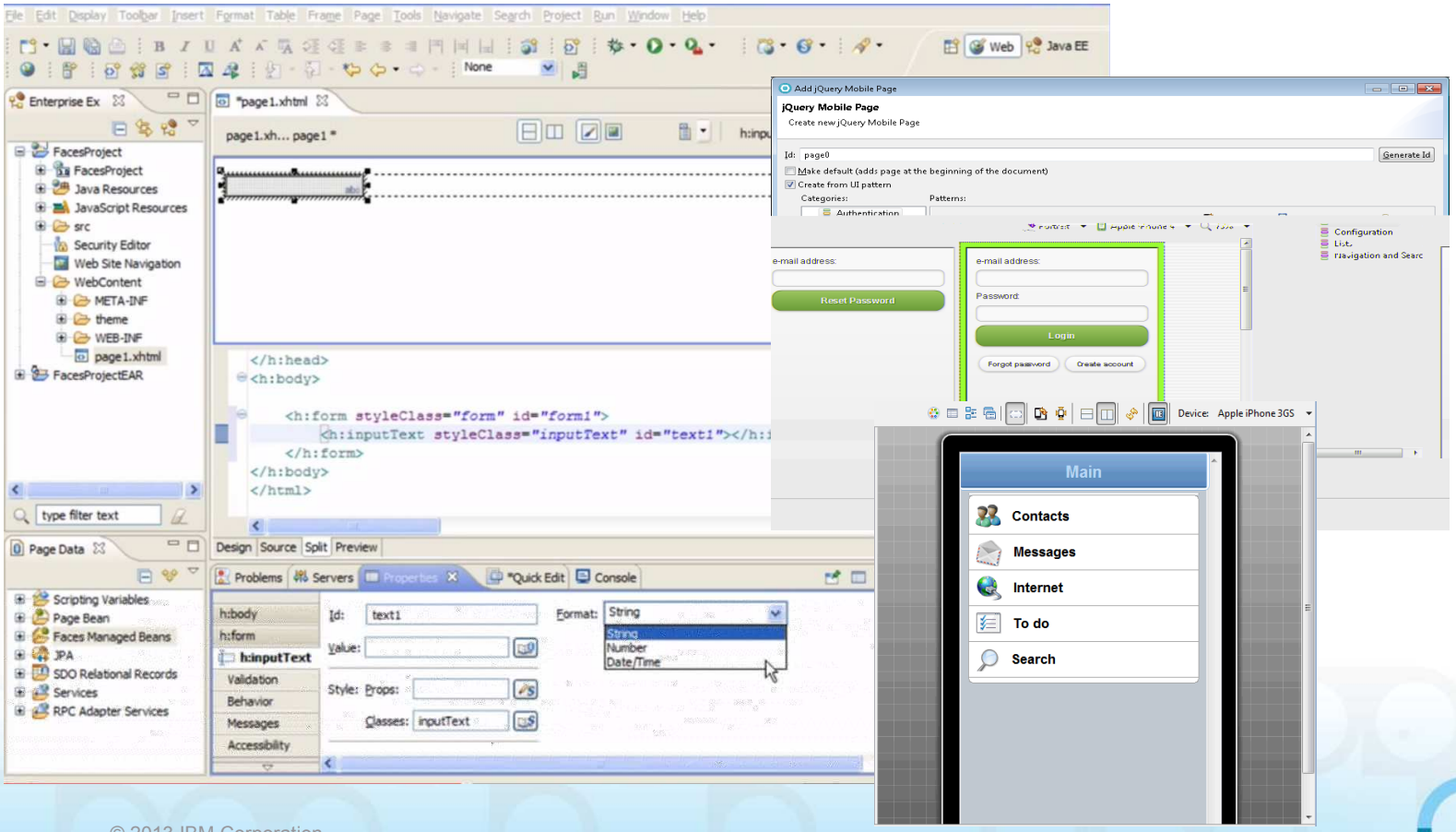
# Typical use cases for WEF

- Any type of application UI, with server-side data access support
- Forms, views, charts, multi-page applications, etc., etc.
- Client-side or server-side applications or combination of both




© 2013 IBM

**Rational Application Developer (RAD)** *provides for Java-based portlets in an Eclipse-based IDE with a comprehensive tool set featuring wizards and Drag/Drop*




© 2013 IBM Corporation

# The RAD approach

- Provides complete tools for the Java developer
  - All the Java/JEE/web tools you need, in a single IDE
  - Tools for UI
  - Tools for REST/Web services
- Wizards, specialized editors, and drag/drop tools are available
- WYSIWYG page editing and mobile browser simulator let you visualize what you're building
- Frameworks such as JSF and Spring can be used
- Developers can publish directly to local or remote server

© 2013 IBM Corporation

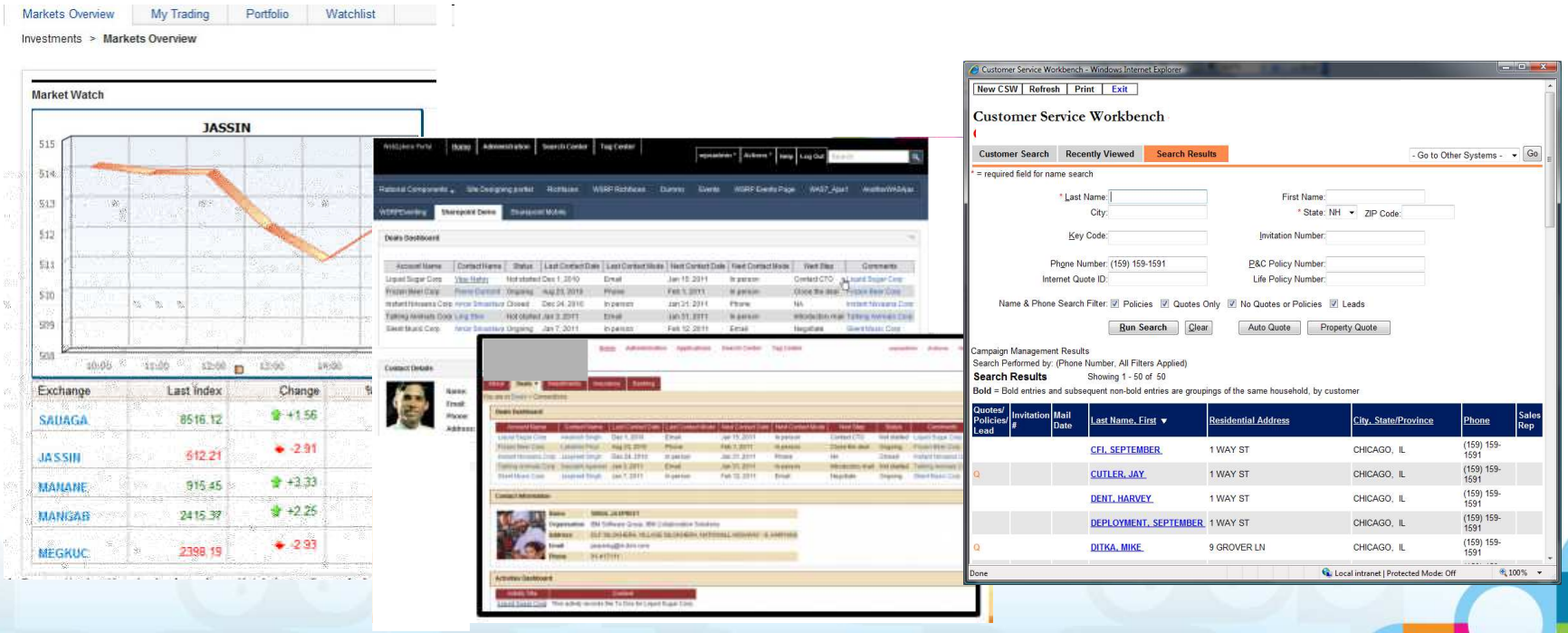
# Key benefits of the RAD approach

- All-in-one IDE gives developers everything they need
- Valuable open source frameworks such as JSF and Spring can be used
- A combination of server-side and client-side code can be used
- Test servers are included
- The Eclipse-based environment with a focus on portal programming is intuitive for many developers
- Specialized tools like Rich Page Editor and Mobile Browser Simulator for mobile development are available
- Code generation using wizards enhances productivity
- Tools support easy integration with JavaScript libraries such as jQuery Mobile or Dojo and with 3rd party JSF component libraries

© 2013 IBM Corporation

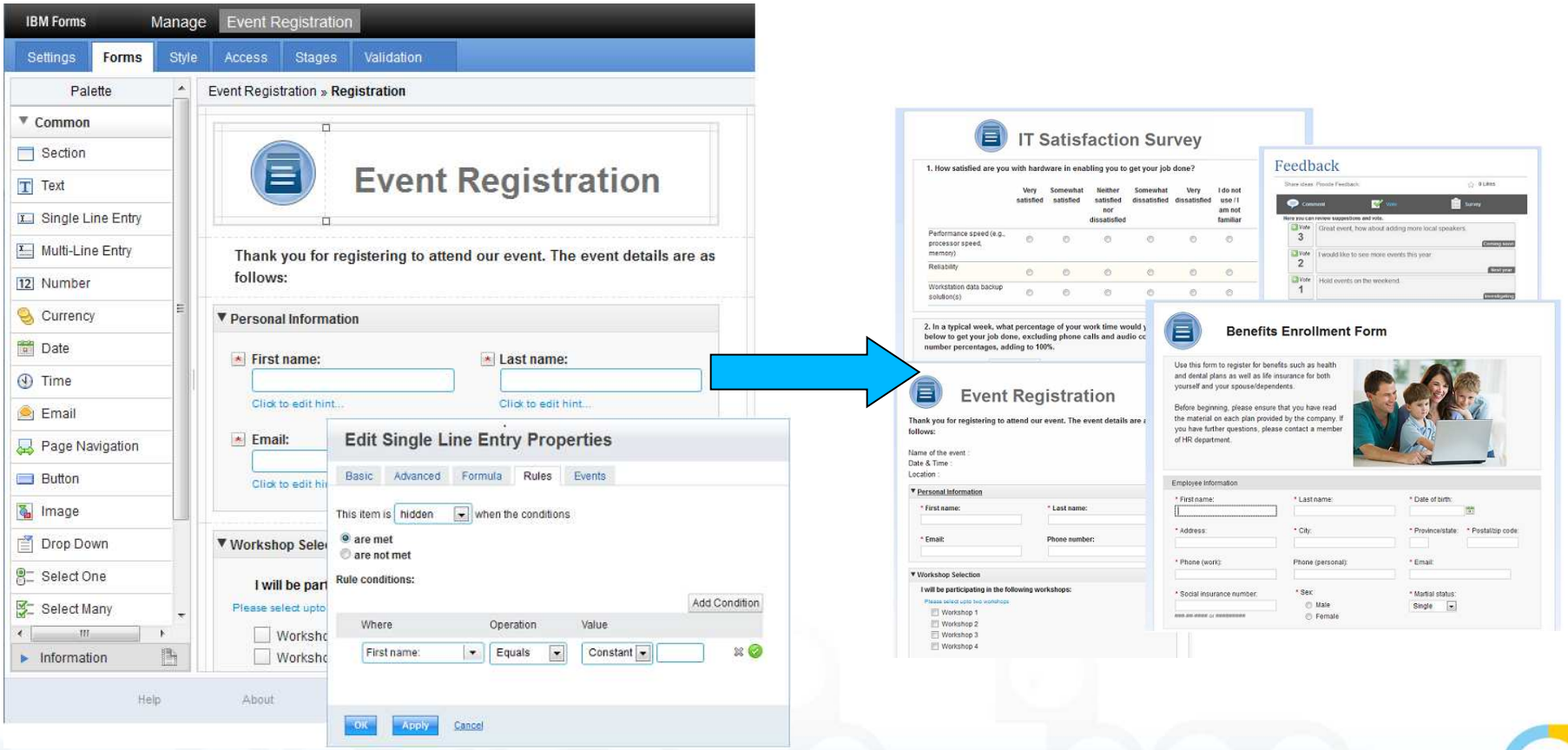
# Typical use cases for RAD

- Any type of application UI, with server-side data access support
- Forms, views, charts, multi-page applications, etc., etc.
- Client-side or server-side applications or combination of both




© 2013 IBM Corporation

**Forms Experience Builder (FEB)** *Build complete data entry form solutions in minutes right from the browser without any coding*




© 2013 IBM Corporation

# The XFEB approach

- Forms are built in a browser with simple WYSIWYG design tools
- The complete application definition is stored in a database
- Applications can be extended and enhanced with JS/HTML/CSS
- Forms data can be stored in a database, or you can use the service architecture to integrate with other applications

© 2013 IBM Corporation

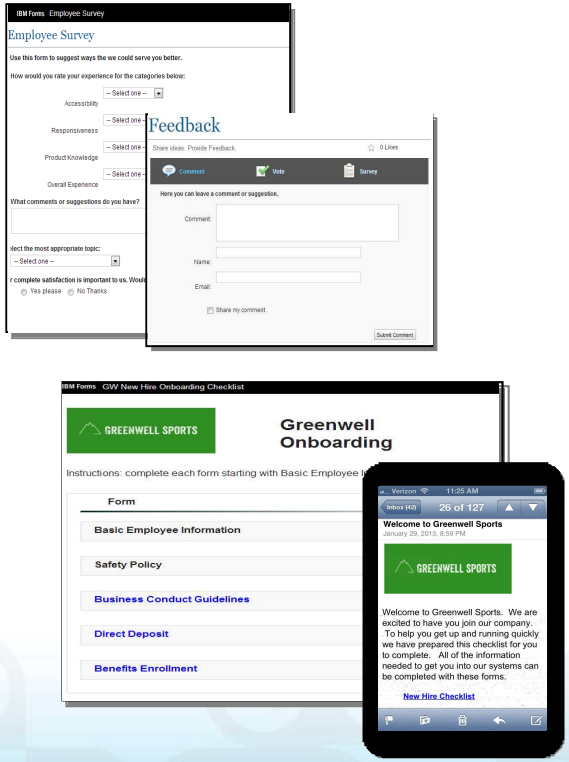
# Key benefits of the FEB approach

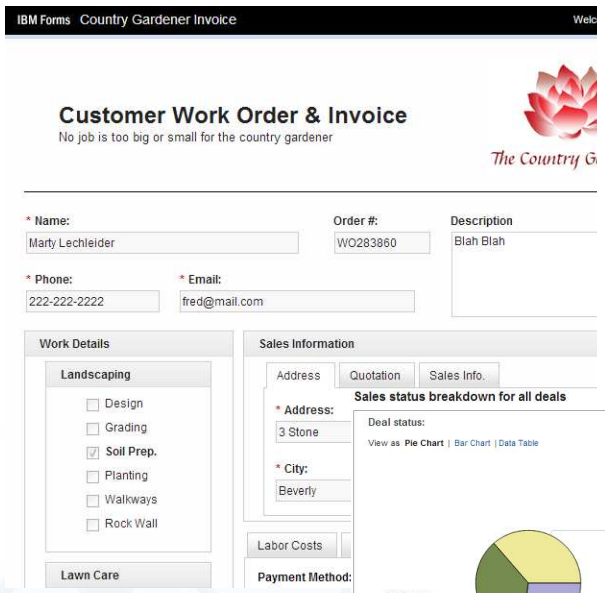
- Non-technical users can build sophisticated data entry form applications
- You can make changes and fine tune form experiences in minutes
-

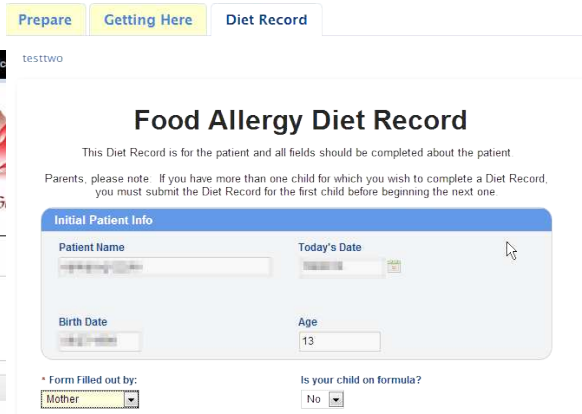
© 2013 IBM Corporation

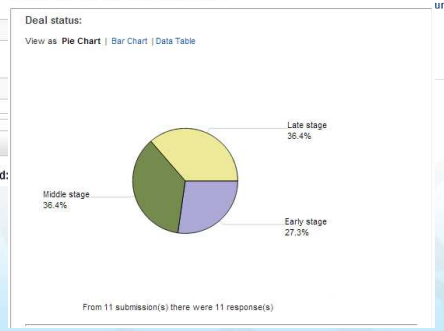
# Typical use cases for FEB

- All kinds of data entry forms
  - Survey, enrollment, feedback, self-service, request, application, registration
- Reports and charts of results




© 2013 IBM Corporation

# Final comments and links to resources

© 2013 IBM Corporation

# Some final comments

- This deck presents a simplified view, and in many places used words such as “primary” or “typical”
- When you include all the things you can do with these tools, the distinctions can seem blurred
- All the tools can do things outside of their most common use cases
- For example:
  - With WEF, RAD, or Eclipse, you can have a small server-based shell and build most of the application with JavaScript and JavaScript libraries and frameworks
    - The WEF Script Application builder automates this pattern
  - While DDC is most often used for rendering, the technology also supports some CRUD applications using REST services

© 2013 IBM Corporation

# For more information

- Script Portlet
  - Script Portlet on Portal Catalog
- Digital Data Connector and Social Rendering
  - Digital Data Connector documentation
  - Blog with slides: IBM Digital Data Connector for WebSphere Portal Now Available
- Web Experience Factory
  - http://www-10.lotus.com/ldd/pfwiki.nsf
- Rational Application Developer Portal Tools
  - RAD Portal Tools wiki
- Forms Experience Builder
  - http://www-10.lotus.com/ldd/lfwiki.nsf/

© 2013 IBM Corporation